



Express Mail Label No. _____

Dated: _____

Docket No.: 03343/000I048-US0
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
Patrick Kerpan

Application No.: 09/755,955

Confirmation No.: 5180

Filed: January 5, 2001

Art Unit: 2192

For: TEMPORAL CONTEXT PROGRAMMING IN
OBJECT-ORIENTED ENVIRONMENTS

Examiner: C.O. Kendall

APPELLANTS' BRIEF ON APPEAL UNDER 37 C.F.R. § 1.192

MS Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

This is Appellant's brief under 37 C.F.R. §1.192 for the appeal of the final Office Action mailed April 8, 2005 by Examiner Chuck O. Kendall in the above-identified patent application.

This brief is filed in furtherance of the October 11, 2005 Notice of Appeal filed in this case. Concurrently, herewith, Appellant submits a Petition for Extension of Time Pursuant to 37 C.F.R. § 1.136(a) requesting an extension of 5 months in which to submit this Appeal Brief. The requisite fee under 37 C.F.R. § 1.17(a) is also enclosed. With such petitions and fee, this Appeal Brief is due on or before May 11, 2006, and thus, is timely.

I. REAL PARTY IN INTEREST

The real party in interest for this appeal is Borland Software Corporation. Rights in and to this application have been assigned to this party.

{W:\03343\000i048-us0\00734435.DOC [*03343000i048-US0*] }

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

There are 14 claims pending in application, i.e. claims 1-14. They are reproduced in the **Claims Appendix**. The current status of the claims is as follows:

1. Claims canceled: none;
2. Claims withdrawn from consideration but not canceled: none;
3. Claims pending: 1-14;
4. Claims allowed: none;
5. Claims rejected: 1-14.

Claims 1-14 stand rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,862,325 to Reed et al. ("Reed"). Claims 1-14 are pending in the application, with all claims on appeal. Appellant submits that Claims 1-14 stand and fall together. The reasons for patentability are set forth below.

IV. STATUS OF AMENDMENTS

In response to the final Office Action mailed April 8, 2005, Appellants filed a Request for Reconsideration on June 7, 2005. The Request for Reconsideration traversed the rejection by presenting arguments distinguishing the pending claims over the cited art, without making any amendments. The Examiner responded thereto in an Advisory Action mailed October 7, 2005. In the Advisory Action, the Examiner indicated that the Request for Reconsideration did not place the application in condition for allowance and entered the submission for purposes of appeal.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present invention is directed to a programming system for managing data entities or objects that manage time-structured data, with a seamless ability to persist in an industry standard relational database. (Specification, page 1, lines 16-19.) More particularly, the present invention is directed to an object-oriented temporal context programming system, where data objects have at least one attribute and are defined by a class of object. (Specification, page 5, lines 3-4.) Such attributes are at least relatively persistently stored in a database. (Abstract) Specifically, the past and present values of the attribute are stored in the database with an indication of the effective time of each value of the attribute. (Abstract). Additionally, methods which the classes can carry out are stored in a similar fashion. (Abstract). Such a system creates a unified way and common programming model for use by application developers in managing temporal data information and point of view information. (Specification, page 4, lines 11-12.) By adding a temporal context in object oriented programming environments, the present invention provides a generic mechanism for managing temporal data in object oriented environments, thus alleviating a programmers need to create a mechanism for managing time structured data and simply requiring the programmer to learn how to use the temporal context concept. (Specification, page 4, lines 16-20.)

The temporal context programming concept is implemented by a structural change of objects, for instance by associating with data objects its attributes at various points in time. (Specification, page 12, lines 11-13.) Moreover, according to the present invention, over time a data object can have a change in type or change in the value of a field. Changes in type are stored in the

database along with a temporal indication. Similarly, when a value changes, the value is saved in the database along with a temporal indication. The present invention highlights that objects are structured such that the attribute value alone is saved in the database and every value change does not result in the creation and storage of a completely new object. (Specification, page 14, lines 8-10).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether Claims 1-14 were erroneously rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,862,325 to Reed et al. ("Reed").

VII. ARGUMENTS

Claims 1-14 Are Patentable Over Reed.

The claimed invention is directed to a computer program for managing data entities or objects that manage time structured data. The present invention introduces a generic mechanism for managing temporal data in object-oriented environments and thus, allows application developers to manage time or temporal data information and point of view information in a unified manner. The present invention easily distinguishes over Reed because Reed fails to disclose or suggest numerous features of the claimed invention and teaches away from the present invention.

The present invention creates standardized means for managing temporal data in object oriented programming environments by adding a context to the environment. Specifically, the present invention enables the storing of multiple versions, or states of the attributes and methods of an object, in a database with an indication of the effective time, thus providing access to past information, as well as enabling effortless management of it. Through the use of effective time, temporal problems are handled in standard ways by the system, and retrieval of data is merely a query of the database.

In contrast to the present invention, Reed does not disclose, nor is Reed concerned with, managing temporal data by using effective time in the retrieval process. Rather, Reed discloses a communications system which coordinates the transfer of various data and instructions between multiple distributed databases, in order to control and process communications and changes to distributed objects. Reed specifically addresses the problem of creating a communication system which allows providers and consumers to establish an automated communication relationship in which the data necessary for operation is exchanged and updated automatically for both parties. Reed is an updating system whose overall goal is to synchronize the information contained in the provider and consumer computers, to match at all times.

The Examiner alleges that Reed discloses the notion of effective time. However, the Appellant strongly disagrees with the Examiner's position. In accordance with its synchronization objective, Reed generally discloses overwriting older versions of attributes, when an attribute of an object is updated, and storing new values along with the modification date. In the instances where Reed allows for the possibility of storing multiple versions of an object in a database, Reed does not broach doing so using effective time. For instance, Reed allows the storage and retrieval of older versions if it is known at the time of the database's creation that it would be necessary to track older attributes. However, in this instance, the program would need to be modified to create reports of any changes in the attributes. Retrieval of past values would not be possible based on the effective time and thus would not be merely a query of the database, as in the present invention. In another instance, where Reed allows for the possibility of retaining several versions, Reed does so based on a number of versions that have previously been stored. Specifically, if a preferred number of previous communications objects are already stored in the database, the archive rule deletes the oldest version; otherwise a new version is added to any preexisting versions. (Reed, column 30, lines 56-61). Such language clearly indicates that the storage and deletion of older versions is based on version numbers and not the specified effective time. Additionally, while objects may not be immediately discarded, Reed clearly asserts that they do not automatically exist while undergoing various changes and modifications.

Effective time is an essential concept of the present invention because when an object undergoes a change in type or value, the changes may be stored along with an indication of the period of time for which that type or value was in effect. Specifically, when a type change occurs, the type is stored in the database with an indication of its effective time. When a value change occurs, the value, and only the value, is stored in the database along with a temporal indication, such that every attribute value change does not result in the creation and storage of a completely new object. In this manner, information is efficiently stored in the database, and a programmer is further enabled simply to provide the effective time to a specified function call and the temporal problem is handled in a standard way, by operating in the same way it would have at that period of time, even if subsequent changes have been made.

In the Advisory Action dated October 7th, 2005, the Examiner attempts to argue that Reed discloses the effective time notion. To support his proposition, the Examiner cites column 90, lines 35-67 and column 92, lines 25-40 of Reed. However, like Examiner's previous citations, such excerpts do not disclose effective time. The cited portion of column 90 specifically discusses how an archiving attribute and rule can be used to control the *number* of previous versions of a communication object that will be archived. It also states that archiving rules allow control of archiving using time intervals.

The Examiner misinterprets the functionality of archiving rules and the use of time intervals. Archiving rules have been previously cited by the Examiner and adequately addressed by the Applicant. Archive rules are specifically directed at a preferred number of previous versions of a communication object, and are used to determine if that number has been exceeded. (Reed, column 30, lines 56-61; column 40, lines 27-32). To state the issue more concretely, old versions of an object can remain on a system only for so long as a predetermined number of versions has not been exceeded. Once a predetermined number of versions is reached, an older version will be overwritten/deleted when a new one is stored. As previously discussed, archiving rules do not disclose effective time, nor do they allow changes to be stored along with an indication of the period of time for which that type or value was in effect.

Similarly, time intervals do not equate to a disclosure of effective time. Time intervals, as discussed in Reed, are merely referred to in disclosing that deleted instances may be maintained “for a specified interval of time before purging them completely from the database.” (Col. 31, ll. 6-8). This language clearly indicates that “time intervals” are simply pre-defined periods of time, and not an indication of the period of time for which a type or value of an object was in effect, which is the proper scope of the notion of effective time. The mention of a time interval, under the heading “Data Archiving Control” in column 90, does not change the scope of its meaning. The Examiner’s citation simply points to wording which states that archiving rules (version numbers) allow control of archiving based on time intervals, which results in only a specified number of older versions being maintained for a predetermined amount of time. In fact, Reed, further discussing Data Archiving Control, specifically discloses that “a rule 140 can specify that any logged event instances 117 older than X interval be deleted, preventing an unnecessary buildup of data.” (Col. 90, ll. 62-67.) It is, in fact, this “buildup of data” that the claimed invention seeks to preserve for “time structured or other context related data.” (Specification, page 4, lines 11-12.)

The Examiner has also misunderstood the different meaning of “persistent” in the context of Reed and the context of the claims. The Examiner cites Reed, column 59, lines 5-15 as disclosing the persistence of the claimed invention. However, this passage merely discloses that “every communications object that will be persistent in the consumer database needs to be updated when the provider makes changes to the object.” (Reed, column 59, lines 5-7.) Thus, Reed uses “persistent” to indicate that an object itself exists for a period of time and is not immediately discarded after use. Moreover, the same passage cited by the Examiner, continues to note that each object that persists in the consumer database, “needs to be updated when the provider makes changes to the object.” Thus, the object will change over time, and previous states or values of the object will be lost when the update occurs, unless the object is archived as a completely new instance of the object. The present application attaches a significantly different meaning to the term “persistent.” Specifically, the present invention uses “persistent” in “[a]n object oriented temporal context programming system,” as recited by claim 1. In this context persistence does not merely refer to the existence of an object for a prolonged period of time, but rather refers to the persistence

of temporal values of past object member data and methods as they change over time, or “current and past states of objects.” (Specification, page 4, lines 2-6.) While the objects of the present invention persist over time, a more salient feature is that each attribute is “relatively persistently stored in the database so that past and present values of the attribute are stored . . . with an indication of the effective time of each value of the attribute,” as recited by claim 1. Reed does not disclose the persistence of attributes, such that “past and present values of the attribute are stored.”

Furthermore, Reed does not disclose persistent “methods which the class can carry out . . . said method being at least relatively persistently stored in the database so that past and present versions of the method are stored in the database with an indication of the effective time of each version of the method,” as recited by claim 1. The Examiner contends that column 69, lines 10-15 of Reed discloses persistent methods, “where Reed discloses allowing a user to dynamically generate persistently stored provider-specific data type definitions in the consumer database.” (Final Office Action, page 3, Aug. 23, 2004.) Provider-specific data types are not the same as methods. Claim 1 recites that a “method being at least relatively persistently stored” can be executed “with a particular time argument utilizing the particular values of attributes of the effected data objects.” In contrast, Reed discloses only data-type definitions such as “a consumer’s preference between a provider’s selection of product colors.” (Reed, column 69, lines 14-16.) Thus, Reed merely discloses storage of data and does not disclose the storage of persistent methods as recited by the present invention.

The Examiner further contends that Reed discloses the use of date/time to provide persistence of “past and present values” of the attributes “stored in the database with an indication of the effective time of each value of the attribute,” as recited by claim 1. However, Reed does not disclose the association of date/time metadata with attributes to provide access to past and present values. Rather, Reed is merely concerned with the synchronization and consistency of multiple instances of the same object distributed on difference computers over a network. Specifically, the passage of Reed cited by the Examiner as disclosing date/time metadata (Reed, column 59, lines 40-59) discusses the use of an “If-Modified-Since” parameter to determine whether a Web server needs to distribute changes in a communications object to a client. For Example, a client can utilize an

“If-Modified-Since” parameter to inform the Web server of the client’s version of the object. If the Web server has a newer version than the client, the Web server will transmit an object markup file (the changes to the object), otherwise the Web server will transmits a “no change” response code. (Reed, column 59, lines 45-59.) In contrast, the claimed invention utilizes the “indication of the effective time” associated with each attribute to provide significantly different and more extensive functionality. The claimed invention is not merely limited to the synchronization of objects over a network using “If-Modified-Since” metadata. Users and developers are able to effectively go back in time and examine or utilize past or present states based on the effective time associated with those states without examining “If-Modified-since” metadata.

The Examiner cites Reed, column 30, lines 48-52, as disclosing an “attribute being at least relatively persistently stored in the database so that past and present values of the attribute are stored in the database with an indication of the effective time of each value of the attribute,” as recited by claim 1. However, the cited passage of Reed discloses the significantly different feature that “multiple versions of object instances may be maintained in the database so that the user can revert to previous data.” (Reed, column 30, lines 49-52.) Reed requires multiple versions of objects to be maintained in the database. Thus, for each change to a single object attribute, the entire object must be replicated in the database. This requires significantly more memory, storage, and processing than the present invention. Additionally, the same object reference can not be used to retrieve past data. The claimed invention maintains only one object, and persistently and efficiently stores the values of the attributes. Thus, the object does not need to be replicated with each change to an individual attribute.

Furthermore, the combination of persistence at the attribute level and the notion of effective time provide significant new and expanded functionality over Reed. For example, in the claimed invention, a user or developer can easily select data points and execute operations using those data points over varying “effective times.” Using a single object instance and specifying various “effective times,” a user can execute a version of a method as it existed in August 2005 using a combination of data that existed one month ago and data that existed three months ago. In contrast, performing a similar task in Reed’s system would require the determination of which

determination of which versions of the object stored in the database correspond to the varying time periods of interest, accessing the instances of those objects associate with the determined versions, and, assuming each required instance is available and has not been deleted by an archiving rule, executing the method of the determined instance of the object using the data of the other determined instances of the object. Thus, the system of the claimed invention provides for significantly advanced and easier manipulation and exploration of temporally persistent data that is not provided by Reed.

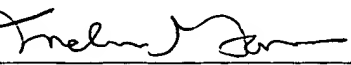
In view of the foregoing, Appellant respectfully submits that Claims 1-14 are not anticipated by Reed. Therefore, it is respectfully submitted that, based upon the preceding discussion, the Examiner should be reversed.

For all of the reasons set forth above, Appellants respectfully request that the application be remanded to the Primary Examiner with an instruction to withdraw the 35 U.S.C. § 102(b) rejection, and pass the case to allowance.

Please charge any fee, except for the Issue Fee, that may be necessary for the continued pendency of this application to our Deposit Account No. 04-0100.

Dated: May 8, 2006

Respectfully submitted,

By 

Melvin C. Garner

Registration No.: 26,272

DARBY & DARBY P.C.

P.O. Box 5257

New York, New York 10150-5257

(212) 527-7700

(212) 527-7701 (Fax)

Attorneys/Agents For Appellants

APPENDIXES

CLAIMS APPENDIX

1. An object-oriented temporal context programming system comprising:
a database;
data objects, each data object being defined by a class of object and having at least one attribute, said attribute being at least relatively persistently stored in the database so that past and present values of the attribute are stored in the database with an indication of the effective time of each value of the attribute; and
methods which the class can carry out, said methods having an argument with an effective time, said method being at least relatively persistently stored in the database so that past and present versions of the method are stored in the database with an indication of the effective time of each version of the method, execution of said method with a particular time argument utilizing the particular values of attributes of the effected data objects and the particular version of the method in effect for the particular time specified.

2. An object-oriented temporal context programming system comprising:
a database;
data objects, each data object being defined by a class of object and having at least one attribute, said attribute being at least relatively persistently stored in the database, so that past and present values of the attribute are stored in the database with an indication of the effective time of each value of the attribute; and

methods which the class can carry out, said methods having an argument which is effective time, execution of said method with a particular time argument utilizing the values of the attributes of the effected data objects in effect for the particular time specified.

3. An object-oriented temporal context programming system comprising:

a database;

data objects, each data object being defined by a class of object and having at least one attribute, said attribute being at least relatively persistently stored in the database, so that past and present values of the attribute are stored in the database; and

methods which the class can carry out, said methods having an argument which is effective time, said method being at least relatively persistently stored in the database, so that past and present versions of the method are stored in the database with an indication of the effective time of each version of the method, execution of said method with a particular time argument utilizing the particular version of the method in effect for the particular time specified.

4. An object-oriented context programming system comprising:

a database;

data objects, each data object being defined by a class of object and having attributes, at least one attribute of one data object being at least relatively persistently stored in the database, so that at least two values of the attribute are stored in the database, each value being associated with an indication of the context thereof; and

methods which the class can carry out, at least one of said methods having an argument which is an indication of context, said method being at least relatively persistently stored in the database, so that at least two versions of the method are stored in the database, each version being associated with an indication of the context thereof, a method executed with a particular context argument utilizing the values of the attributes of the effected data objects and the version of the method in effect for the particular context.

5. An object-oriented context programming system as claimed in claim 4 wherein the context is a version of an application program, so that by identifying a particular context a different version of the application program runs and gives the user a different vantage point from which to experience the program.

6. An object-oriented context programming system comprising:
a database;
data objects, each data object being defined by a class of object and having attributes, at least one attribute of one data object being at least relatively persistently stored in the database, so that at least two values for the attribute are stored in the database, each value being associated with an indication of the context of the value; and

methods which the class can carry out, at least one of said methods having an argument which an indication of context, a method executed with a particular context argument utilizing the values of the attributes of the effected data objects in effect for the particular context.

7. An object-oriented context programming system comprising:
data objects each being defined by a class of object and having attributes; and
methods which the class can carry out, at least one of said methods having an
argument which is an indication of context, said method being at least relatively persistently stored
in the database, so that at least two versions of the method are stored in the database each version
being associated with an indication of the context of the method, a method executed with a
particular context argument utilizing the version of the method in effect for the particular context.

8. An object-oriented temporal context programming system as claimed in any one of
claims 1- 3, further including a new attribute added to said data object and being stored in the
database with an indication of the effective time of the new attribute, which effective time is
subsequent to the time of creation of the object.

9. An object-oriented context programming system as claimed in any one of claims 4 -
7, further including a new attribute added to said data object and being stored in the database with
an indication of the context of the new attribute.

10. An object-oriented temporal context programming system as claimed in any one of
claims 1- 3, wherein the execution of said method is with respect to a time in the past.

11. An object-oriented temporal context programming system as claimed in claim 10
wherein one attribute has an additional context of an error and an equivalent attribute has an

additional context of the error corrected, and wherein the methods can be run to show the effect in the past both with and without the error.

12. An object-oriented temporal context programming system as claimed in any one of claims 1- 3, wherein the execution of said method is with respect to a time in the future, and the execution of the methods predicts events in the future based on probabilities.

13. An object-oriented temporal context programming system as claimed in any one of claims 1- 3, wherein said class of object is formed from a temporal base class as a subclass of the temporal base object class and inherits its temporal context capabilities of reading (getting) or storing (setting) from the temporal base object class.

14. An object-oriented context programming system as claimed in any one of claims 4-7, wherein said class of object is formed from a base object class as a subclass of the base object class which inherits its context capabilities of reading (getting) or storing (setting) from the base object class.

Application No.: 09/755,955

Docket No.: 03343/000I048-US0

EVIDENCE APPENDIX

All evidence is in the record.

Application No.: 09/755,955

Docket No.: 03343/000I048-US0

RELATED PROCEEDINGS APPENDIX

None.